HOCHSCHULE MITTWEIDA

UNIVERSITY OF APPLIED SCIENCES

Generierung von Protein-Protein-Interaktionsnetzwerken unter Verwendung einer t-Conorm und Finden von Interaktionswegen

eingereicht als

Bachelorarbeit

Thielemann, Katharina

Seminargruppe MA10w1-B

Angewandte Mathematik

MNI

01.12.2013

Betreuer Fakultät MNI: Prof. Dr. rer. nat. Dirk Labudde
Betreuer Fakultät MNI: Prof. Dr. rer. nat. habil. Thomas Villmann

Danksagung

Zunächst möchte ich mich an dieser Stelle bei all denjenigen bedanken, die mich während der Anfertigung dieser Bachelorarbeit unterstützt und motiviert haben.

Ein besondere Dank geht dabei an Herrn Prof. Dr. Thomas Villmann und Herrn Prof. Dr. Dirk Labudde, sowie an Stefan Schildbach.

Ein ganz besondere Dank gilt ebenfalls meiner Mutter, meinen Großeltern und meinem Freund Till Martin Riedel, denn sie standen mir nicht nur finanziell, sondern auch emotional während meines gesamten Studiums zur Seite.

Katharina Thielemann

Inhaltsverzeichnis

1 Einleitung

Protein-Protein-Interaktionen spielen eine wichtige Rolle bei allen zellulären Prozessen. Dazu zählen unter anderem Transportprozesse innerhalb einer Zelle in einem Organismus, welche mit Hilfe von Proteinkomplexen vollzogen werden. Ein weiteres Beispiel ist das Zytoskelett. Hier befinden sich sogenannte Aktinfilamente, die in der Zellmembran eingelagert und verantwortlich für die Stabilität der Zelle sind. Aktinfilamente bestehen aus Proteinkomplexen, welche durch Proteininteraktionen zusammengehalten werden [9].

Auch aus medizinischer Sicht ist das Verstehen von Proteininteraktion von großer Bedeutung. So spielen beispielsweise fehlregulierte Interaktionen eine entscheidende Rolle bei Erkrankungen wie Krebs oder auch Alzheimer [10].

Eukaryotische Zellen enthalten hunderte Proteinkomplexe, wobei eine Vielzahl solcher Komplexe wiederum aus hunderten von Proteinen bestehen kann. Man kann sich also die Proteine einer Zelle als Knoten eines riesigen Proteinnetzwerkes vorstellen.

Der menschliche Körper besitzt circa 25.000 Proteine, welche durch 100.000 Interaktionen miteinander verbunden sind. Einige 10.000 solcher Interaktionen sind bereits identifiziert und in Datenbanken abgelegt.

Um Proteininteraktionen zu erforschen, existiert eine Vielzahl von bestimmten Methoden und Experimenten. Diese Experimente variieren jedoch sehr stark in ihrer Zuverlässigkeit. So ist beispielsweise gezeigt worden, dass die Technik *Yeast Two-Hybrid* eine große Rate an falsch positiv ermittelten Interaktionen aufzeigt [14].

Gegenstand dieser Arbeit soll es sein, die Daten für Protein-Protein-Interaktionen aus verschiedenen Datenbanken miteinander zu verknüpfen, ein Netzwerk zu modellieren und anschließend in diesem spezielle Wegen zu finden. Dafür

werden in den Abschnitten in Kapitel 2 zunächst die mathematischen Grundlagen geschaffen, sowie ein Einblick in die verwendeten Datenbanken gegeben. Dabei werden kurz auf die Bewertungsverfahren, welche den Proteinen und ihren Interaktionspartnern einen Wert (Score) zuweisen, eingegangen. In Kapitel 3 wird anschließend die Methodik erklärt, auf welche Weise die Informationen verknüpft werden und welche Auswirkung dies auf Wege innerhalb des Netzwerkes hat. Abschließend werden in der Zusammenfassung, neben den wichtigsten Ergebnissen, auch Anregungen und Konzepte für Erweiterungen vorgestellt.

2 Grundlagen

In diesem Kapitel soll sowohl ein kleiner Einblick in die experimentellen Methoden zur Untersuchen von Proteininteraktionen, sowie die mathematischen Grundlagen und die verwendeten Datenbanken, welche in dieser Arbeit Anwendung finden, gegeben werden.

2.1 Protein-Protein-Interaktionen

Nachdem die Sequenzierung des Humangenoms abgeschlossen war, begannen Forscher sich auf die systematische Untersuchung von menschlichen Proteinen zu konzentrieren. Viele Proteine interagieren miteinander um ihre Aufgaben zu erfüllen. Solche Komplexe aus Proteinen bezeichnet man als Protein-Protein-Interaktionen, welche durch sogenannte Van-der-Waals-Kräften, Wasserstoffbrückenbindungen und elektrostatischen Wechselwirkungen hervorgerufen werden.

Ziel der Erforschung von Protein-Protein-Interaktionen ist es, unter anderem, aus dem Wissen über die Funktion eines Proteins Rückschlüsse auch auf die Funktion der interagierenden Proteine zu ziehen, um molekulare Zusammenhänge innerhalb einer Zelle im Organismus zu verstehen.

Für die Erforschung von Interaktionen zwischen Proteinen haben sich eine Vielzahl von unterschiedlichen experimentellen Methode entwickelt. Abbildung 2.1 zeigt eine Auswahl solcher Methoden und die molekulare Ebene der Interaktion, welche man dabei beobachten kann (schwarze Färbung in den Tabellenzellen).

Das obere Bild der Darstellung zeigt hierbei die Bindung eines Liganden (dunkelgrau gefärbter Kreis) an einen Rezeptor (dargestellt als hellgraue Ellipsen). Zunächst lassen sich den verschiedenen Beobachtungsebenen vier

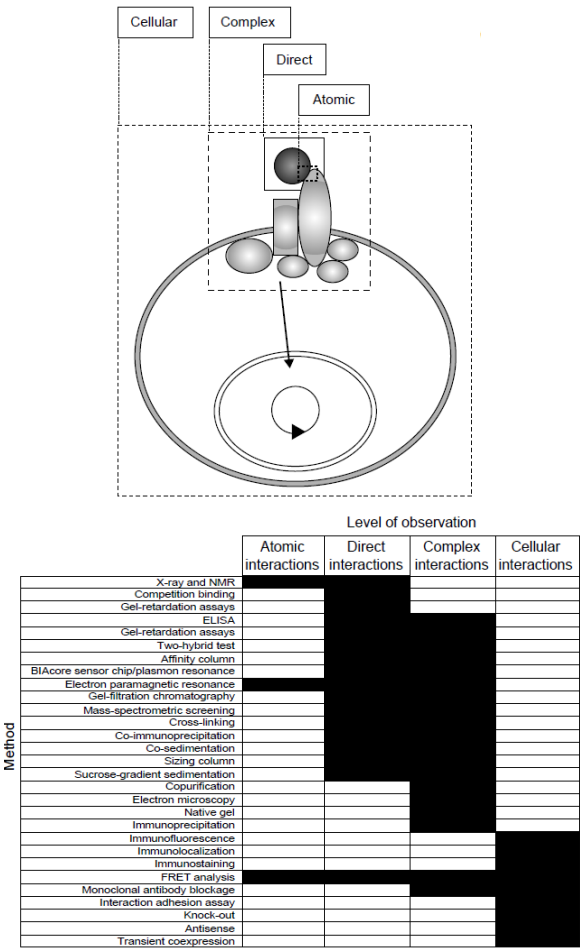


Abbildung 2.1: Einteilung der Interaktionsart in Abhängigkeit der experimentellen Methoden [12]

verschiedenen Interaktionstypen zuordnen: eine Interaktion auf atomarer Ebene, einer direkten Interaktion, die Interaktion eines ganzen Komplexes und zelluläre Interaktionen. Das untere Bild zeigt, mit welchen Methoden sich welche Interaktionsebene untersuchen lässt. So können beispielsweise Experimenten mit Hilfe von Röntgenstrahlen (*X-Ray*) Aufschlüsse darüber geben, welche Atome an der jeweiligen Interaktion beteiligt sind. Hingegen Versuche mittels einem Hefe-Zwei-Hybrid-Systemen (*Y2H*) Nachweise über die an der Interaktion beteiligten Proteine liefern.

2.2 Mathematische Grundlagen

2.2.1 Klassische Mengenlehre

Georg Cantor, welcher als Begründer der Mengenlehre gilt, definierte eine Menge wie folgt: *„Eine Menge ist eine Zusammenfassung von bestimmter, wohl unterschiedener Dinge unserer Anschauung oder unseres Denkens, welche Elemente der Menge genannt werden, zu einem Ganzen.“* [2]
Im Folgenden werden Mengen mit Großbuchstaben und Elemente einer Menge mit Kleinbuchstaben bezeichnet.
Mengen lassen sich auf verschiedene Arten darstellen. Neben der expliziten Aufzählung aller enthaltenen Mengenelemente, besteht ebenfalls die Möglichkeit Mengen mithilfe von charakteristischen Prädikaten anzugeben. So lässt sich beispielsweise die Menge aller geraden natürlichen Zahlen wie folgt angeben:

$$A = \{2, 4, 6, 8, 10, 12, ...\} \tag{2.1}$$

$$A = \{n \in \mathbb{N} \mid n\ gerade\} \tag{2.2}$$

Mengen, wie A aus 2.2, lassen sich allgemein folgendermaßen formulieren:

$$A = \{x \in X \mid P(x)\}, \tag{2.3}$$

wobei $A \subseteq X$ und P das für A charakteristische Prädikat ist. Das heißt A besteht aus den Elementen x einer Menge X, welche das Prädikat P erfüllen. Im Folgenden nennen wir die Menge X die Grundmenge, über welche A definiert ist.

Eine Weitere Darstellungsform einer Menge A über einer Grundmenge X besteht in der Angabe einer charakteristischen Funktion λ.

Definition:
Es sei X eine Grundmenge, $A \subseteq X$ eine Teilmenge von X und $x \in X$.
Dann heißt die Funktion:

$$\lambda_A : X \to \{0, 1\}$$

mit

$$\lambda_A(x) = \begin{cases} 1 & \text{für } x \in A \\ 0 & \text{für } x \notin A \end{cases}$$ ◀

die charakteristische Funktion von A.

Interpretiert man die Werte „ 1 “ und „ 0 “ aus Definition 1 als Wahrheitswerte, d.h. der Wert „ 1 “ entspricht einem „wahr“ und „ 0 “ einem „falsch“, so ist ersichtlich, dass zwischen der Beschreibung einer Menge A mittels eines charakteristischen Prädikates und der mit Hilfe einer charakteristischen Funktion kein relevanter Unterschied besteht. Das Prädikat einer Menge A kann für ein $x \in X$ wahr sein, wenn es die Eigenschaft P von A besitzt, andernfalls falsch. Damit ist jede Menge A durch ihre charakteristische Funktion beschrieben.
Mit Hilfe dieser Funktionen lassen sich nun Teilmengen und Gleichheit von zwei Mengen A und B über derselben Grundmenge X definieren.

Definition: ($\subseteq$, $=$, $\overline{\square}$, $\cap$, $\cup$)

Seien A, B Mengen über der Grundmenge X und λ_A und λ_B die zugehörigen charakteristischen Funktionen. Dann gilt:

$$\text{Teilmenge: } A \subseteq B \Leftrightarrow \lambda_A(x) \leq \lambda_B(x) \tag{2.4}$$

$$\text{Gleichheit: } A = B \Leftrightarrow \lambda_A(x) = \lambda_B(x) \tag{2.5}$$

$$\underline{\text{Komplement:}}\ \overline{A} := \{x|\lambda_{\overline{A}}(x) = 1, x \in X\} \text{ mit } \lambda_{\overline{A}}(x) := 1 - \lambda_A(x) \tag{2.6}$$

$$\underline{\text{Durchschnitt:}}\ A \cap B := \{x|\ \lambda_A(x) = 1 \wedge \lambda_B(x) = 1\} \tag{2.7}$$

$$\underline{\text{Vereinigung:}}\ A \cup B(x) := \{x|\ \lambda_A(x) = 1 \vee \lambda_B(x) = 1\} \tag{2.8}$$

$\forall x \in X.$ ◀

Wie aus der klassischen Mengentheorie bekannt, so gelten auch hier die Eigenschaften der Mengenoperationen und können in [7] nachgeschlagen werden.

2.2.2 Fuzzy Mengenlehre

In Abschnitt 2.2.1 ist gezeigt worden, dass jede Menge, welche über eine Grundmenge definiert ist, durch ihre charakteristische Funktion beschrieben werden kann. Der Wertebereich bestand hierbei aus der zweielementigen Menge $\{0, 1\}$.
Folgendes kleines Beispiel, welches rein zur Illustration dient, soll jedoch zeigen, dass dies zu Diskrepanzen führen kann. Es sei X eine Menge von Autos, $A \subseteq X$ die Menge aller roten Autos von X, d.h:

$$A = \{x \in X \mid x \text{ ist ein rotes Auto } \} \tag{2.9}$$

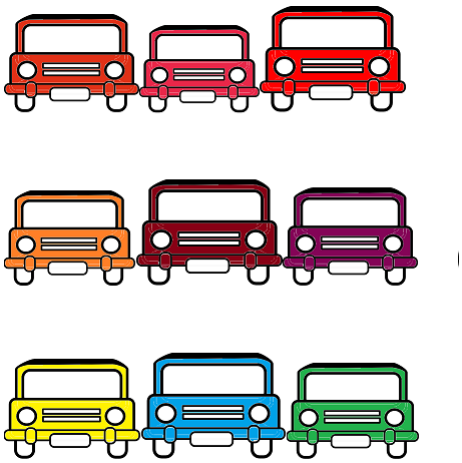


Abbildung 2.2: exemplarische Darstellung von Elementen der Grundmenge X

Abbildung 2.2 zeigt einige Elemente aus der Grundmenge. Die Menge A besteht gemäß Abschnitt 2.2.1 aus der obersten Autoreihe. Betrachtet man das Beispiel jedoch eher intuitiv und würde die Elemente der Menge A angeben, wären dann nicht ebenfalls Autos der zweiten Reihe Elemente der Menge A? Wie bereits oben erwähnt, dient das Beispiel einem rein illustrierenden Zweck. Es soll jedoch symbolisch für verschiedene weitere Probleme stehen, bei denen es sinnvoll ist, den Wertebereich zu erweitern. Das heißt jedem Element eine Grundmenge X wird nicht genau einer der beiden Werte „0 “oder „1 “zugeordnet, sondern stattdessen ein Wert aus dem geschlossenen Einheitsintervall $[0, 1]$. Dieser Gedanke ist zentraler Gegenstand der sogenannten *Fuzzy-Theorie* und wurde vom amerikanischen Mathematiker Lotfi Zadeh 1965 erstmals publiziert [13]. Zadeh erweiterte hierbei den klassischen Mengenbegriff, wie er in 2.2.1 vorgestellt wurde, indem er die Zweiwertigkeit der Elemente auflöste. Hierfür führt er eine Zugehörigkeitsfunktion $\mu_{\tilde{A}}$ ein, welche jedem $x \in X$ einen Zugehörigkeitsgrad $\mu_{\tilde{A}}(x) \in [0, 1]$ zuordnet und definierte eine Menge im Sinne der Fuzzy-Theorie wie folgt:

Definition: (Fuzzy-Menge)

Es sei X ein Grundbereich und $\mu_{\tilde{A}}$ eine Zugehörigkeitsfunktion, die jedem $x \in X$ einen Zugehörigkeitswert $\mu_{\tilde{A}}(x) \in [0,1]$ zuordnet.
Dann heißt die Menge:

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) \mid x \in X\} \tag{2.10}$$

eine Fuzzy Menge. ◀

Im Folgenden werden Fuzzy Mengen stets durch eine Tilde gekennzeichnet.
Zadeh definierte, analog zu den Formeln 2.4- 2.6, das Komplement einer Fuzzy-Menge, sowie Teilmengen und Gleichheit zweier Fuzzy-Mengen, folgendermaßen:

Definition: ($\overline{\square}$, $\subseteq$, = von Fuzzy-Mengen)

Seien $\tilde{A}$, $\tilde{B}$ Fuzzy-Mengen über der Grundmenge X und $\mu_{\tilde{A}}$ und $\mu_{\tilde{A}}$ die zugehörigen Zugehörigkeitsfunktionen. Dann gilt:

$$\text{Komplement: } \overline{\tilde{A}} := \{(x, \mu_{\overline{\tilde{A}}} \mid x \in X\} \text{ mit } \mu_{\overline{\tilde{A}}}(x) := 1 - \mu_{\overline{\tilde{A}}}(x) \tag{2.11}$$

$$\text{Teilmenge: } \tilde{A} \subseteq \tilde{B} \Leftrightarrow \mu_{\tilde{A}}(x) \leq \mu_{\tilde{B}}(x) \quad \forall x \in X \tag{2.12}$$

$$\text{Gleichheit: } \tilde{A} = \tilde{B} \Leftrightarrow \mu_{\tilde{A}}(x) = \mu_{\tilde{B}}(x) \quad \forall x \in X. \tag{2.13}$$

◀

Was jedoch den Durchschnitt und die Vereinigung betreffen, so haben sich in der Fuzzy-Theorie eine Reihe von verschiedenen Operatoren etabliert. Es sollen jedoch in diesem Abschnitt nur die für die Arbeit notwendigen erläutert werden.

Zunächst sei $\tilde{I} = \tilde{A} \cap \tilde{B}$ der Durchschnitt zweier Fuzzy-Mengen $\tilde{A},\ \tilde{B}$ und $i(\mu_{\tilde{A}}(x),\ \mu_{\tilde{B}}(x))$ die zugehörige Abbildung mit

$$i : X \times X \ \rightarrow \ X \tag{2.14}$$

$$(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)) \ \mapsto \ \mu_{\tilde{C}}(x) \tag{2.15}$$

$\forall x \in X$.
Da klassische Mengen spezielle Fuzzy-Mengen sind, muss für alle Durchschnittsabbildungen folgendes gelten:

$$\mu_{\tilde{C}}(x) = 1 \ \Leftrightarrow \ \mu_{\tilde{A}}(x) = \mu_{\tilde{B}}(x) = 1 \tag{2.16}$$

$$\mu_{\tilde{C}}(x) = 0 \ \Leftrightarrow \ \mu_{\tilde{A}}(x) = 0 \ oder \ \mu_{\tilde{B}}(x) = 0 \tag{2.17}$$

$$\mu_{\tilde{A}}(x),\ \mu_{\tilde{B}}(x) \in [0,1] \ \Rightarrow \ \mu_{\tilde{C}}(x) \in [0,1] \ \ \forall x \in X \tag{2.18}$$

$$i(0,1) = i(1,0) \ = \ i(0,0) = 0. \tag{2.19}$$

Abbildungen, welche zur Durchschnittsbildung benutzt werden, nennt man t-Normen und sind wie folgt definiert.
Definition: (t-Norm)

Es seien X eine Grundmenge, $\tilde{A}$, $\tilde{B}$ und $\tilde{C}$ Fuzzy-Mengen über X mit $\mu_{\tilde{A}}(x)$, $\mu_{\tilde{B}}(x)$, $\mu_{\tilde{C}}(x)$. Eine Abbildung $T : [0,1] \times [0,1] \rightarrow [0,1]$ heißt t-Norm, wenn sie folgende vier Axiome erfüllt:

(1) Assoziativität:

$$T(\mu_{\tilde{A}}(x), T(\mu_{\tilde{A}}(x), \mu_{\tilde{C}}(x))) = T(T(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)), \mu_{\tilde{C}}(x)) \tag{2.20}$$

(2) Kommutativität:

$$T(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)) = T(\mu_{\tilde{B}}(x), \mu_{\tilde{A}}(x)) \tag{2.21}$$

(3) Monotonie:

$$\mu_{\tilde{A}}(x) \leq \mu_{\tilde{B}}(x) \Rightarrow T(\mu_{\tilde{A}}(x), \mu_{\tilde{C}}(x)) \leq T(\mu_{\tilde{B}}(x), \mu_{\tilde{C}}(x)) \quad (2.22)$$

(4) Randbedingungen:

$$T(\mu_{\tilde{A}}(x), 1) = \mu_{\tilde{A}}(x), T(\mu_{\tilde{A}}(x), 0) = 0 \quad (2.23)$$

$\forall x \in X.$ ◀

Oft verwendete t-Normen sind beispielsweise:

Minimum:

$$T_M(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)) = min\{\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)\}$$

Algebraische Produkt:

$$T_P(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)) = \mu_{\tilde{A}}(x) \cdot \mu_{\tilde{B}}(x)$$

Beschränkte Summe:

$$T_S(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)) = max\{\mu_{\tilde{A}}(x) + \mu_{\tilde{B}}(x) - 1, 0\}$$

Analog zum Durchschnitt sei nun $\tilde{U} = \tilde{A} \cup \tilde{B}$ die Vereinigung zweier Fuzzy-Mengen $\tilde{A},\ \tilde{B}$ und $u(\mu_{\tilde{A}}(x),\ \mu_{\tilde{B}}(x))$ die zugehörige Abbildung mit

$$u : X \times X \rightarrow X \quad (2.24)$$

$$(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)) \mapsto \mu_{\tilde{U}}(x) \quad (2.25)$$

$\forall x \in X$, wobei für die Vereinigungsoperatoren folgendes gelte:

$$\mu_{\tilde{U}}(x) = 0 \quad \Leftrightarrow \quad \mu_{\tilde{A}}(x) = \mu_{\tilde{B}}(x) = 0 \tag{2.26}$$

$$\mu_{\tilde{C}}(x) = 1 \quad \Leftrightarrow \quad \mu_{\tilde{A}}(x) = 1 \; oder \; \mu_{\tilde{B}}(x) = 1 \tag{2.27}$$

$$\mu_{\tilde{A}}(x),\ \mu_{\tilde{B}}(x) \in [0,1] \quad \Rightarrow \quad \mu_{\tilde{U}}(x) \in [0,1] \ \ \forall x \in X \tag{2.28}$$

$$u(1,1) = u(1,0) \quad = \quad u(0,1) = 1 \tag{2.29}$$

Definition: (t-Conorm)

Es seien X eine Grundmenge, $\tilde{A}$, $\tilde{B}$ und $\tilde{C}$ Fuzzy-Mengen über X mit $\mu_{\tilde{A}}(x)$, $\mu_{\tilde{B}}(x)$, $\mu_{\tilde{C}}(x)$. Eine Abbildung $S : [0,1] \times [0,1] \rightarrow [0,1]$ heißt t-Conorm, wenn sie folgende vier Axiome erfüllt:

(1) Assoziativität:

$$S(\mu_{\tilde{A}}(x), S(\mu_{\tilde{B}}(x), \mu_{\tilde{C}}(x))) = S(S(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)), \mu_{\tilde{C}}(x)) \tag{2.30}$$

(2) Kommutativität:

$$S(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)) = S(\mu_{\tilde{B}}(x), \mu_{\tilde{A}}(x)) \tag{2.31}$$

(3) Monotonie:

$$\mu_{\tilde{A}}(x) \leq \mu_{\tilde{B}}(x) \Rightarrow S(\mu_{\tilde{A}}(x), \mu_{\tilde{C}}(x)) \leq S(\mu_{\tilde{B}}(x), \mu_{\tilde{C}}(x)) \tag{2.32}$$

(4) Randbedingungen:

$$S(\mu_{\tilde{A}}(x), 1) = 1 \text{ und }, S(\mu_{\tilde{A}}(x), 0) = \mu_{\tilde{A}}(x) \tag{2.33}$$

$\forall x \in X.$ ◀

Wie bei der t-Norm, werden hier exemplarisch folgende t-Conormen aufgeführt:

Maximum:

$$S_M(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)) = max\{\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)\}$$

Algebraische Produkt:

$$S_P(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)) = \mu_{\tilde{A}}(x) + \mu_{\tilde{B}}(x) - \mu_{\tilde{A}}(x) \cdot \mu_{\tilde{B}}(x)$$

Beschränkte Summe:

$$S_S(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)) = min\{\mu_{\tilde{A}}(x) + \mu_{\tilde{B}}(x), 1\}$$

Einen weiteren Schwerpunkt dieser Arbeit stellt die Untersuchung von Protein-Protein-Interaktionsnetzwerken dar. Diese können als gewichteter Graph modelliert werden. Die Kantengewichte werden später aus den Methodiken der Fuzzy-Theorie hervorgehen. Dazu sollen im nächsten Abschnitt grundlegende Begriffe der Graphentheorie eingeführt werden, um später die Analyse der PPI-Netzwerke beschreiben zu können.

2.2.3 Graphentheorie

Die Graphentheorie ist eine Teilgebiet der diskreten Mathematik und untersucht Strukturen von Graphen. Ein *Graph* $G = (V, E)$ besteht dabei aus einer *Menge von Knoten* V und einer *Menge von Kanten* E. In der Graphentheorie unterscheidet man zwischen *gerichteten*, d.h. die Kanten weisen eine Orientierung auf, und *ungerichteten* Graphen. Abbildung 2.3 zeigt hierfür jeweils eine Beispiel.

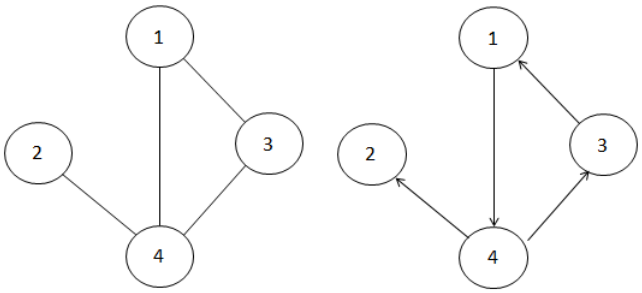


Abbildung 2.3: ungerichteter Graph (links) und gerichteter Graph (rechts)

Zwei durch eine Kante miteinander verbundene Knoten heißen *adjazent*. Außerdem heißen zwei Kanten $e, f \in E$ zueinander *inzident*, wenn sie einen gemeinsamen Knoten $v \in V$ besitzen. Eine Kante heißt *Schlinge*, wenn sie zu sich selbst inzident ist. Zwei Kanten $e, f \in E$ nennt man *parallel*, wenn ihre Knoten übereinstimmen. Daraus abgeleitet ergibt sich der Begriff eines *schlichten Graphen*. Ein Graph, der weder Schlingen noch parallele Kanten enthält, heißt schlicht.

Im Beispiel aus Abbildung 2.3 handelt es sich links um einen ungerichteten, schlichten Graphen mit $V = \{1, 2, 3, 4\}$ und der Kantenmenge
$E = \{\{1, 3\}, \{1, 4\}, \{3, 4\}, \{2, 4\}\}$. Der rechte Graph der Abbildung ist ebenfalls schlicht aber gerichtet.

Definition: (Nachbarschaft eines Knotens)

Es sei $G = (V, E)$ ein Graph mit Knotenmenge V und Kantenmenge E und $v \in V$. Die Nachbarschaft $N[v]$ von v ist:

$$N[v] := \{u \in V | \{u, v\} \in E\} \tag{2.34}$$

◀

Wird jeder Kante $e \in E$ eine reelle Zahl zugeordnet $c(e)$, so spricht man von einem *kantengewichteten Graphen*. Das heißt, es existiert eine Abbildung der Art: $c : E \to \mathbb{R}$.

Definition: (Kantenfolge,Weg)

Es sei $G = (V, E)$ ein Graph. Eine Kantenfolge ist eine Folge von

$$v_1, e_1, v_2, ..., e_{k-1}, v_k \tag{2.35}$$

von Knoten und Kanten, sodass die Kante e_i die Endknoten v_i und v_{i+1} hat.

Die Länge der Kantenfolge ist dabei die Anzahl der Kanten der Folge.

Ein Weg in einem Graphen ist eine Kantenfolge, in der jeder Knoten des Graphen höchstens einmal in der Kantenfolge enthalten ist. ◀

Definition: (Länge eines Weges, Distanz)

Sei $G = (V, E)$ ein Graph und $c : E \to \mathbb{R}$ eine Gewichtsfunktion, die jeder Kante $e \in E$ ein Gewicht $c(e)$ zuordnet. Die Länge $c(P)$ eines Weges $P = (v_0, e_1, v_1, ..., e_k, v_k)$ in G ist definiert durch:

$$c(P) := \sum_{i=1}^{k} c(e_i) \tag{2.36}$$

Die Distanz $dist_c(u, v, G)$ zwischen zwei Knoten $u, v \in V$ bezüglich der Gewichtsfunktion c ist:

$$dist_c(u, v, G) := inf\{c(P) : P\ ist\ Weg\ in\ G\ von\ u\ nach\ v\} \tag{2.37}$$

◀

Graphen lassen sich auf unterschiedliche Weisen darstellen. Üblich ist die Darstellung in Form einer Adjazensmatrix. Die *Adajazenzmatrix* A von einem Graphen G ist eine symmetrische und quadratische $n \times n$ Matrix, wobei n die Anzahl der Knoten des Graphen ist. Die Matrixelemente geben dabei an, ob zwei Knoten i und j adjazent sind, d.h.:

$$a_{i,j} = \begin{cases} 1 & \text{, falls } (i, j) \in E \\ 0 & \text{, sonst .} \end{cases} \tag{2.38}$$

Für den ungerichteten Graphen aus Abbildung 2.3 hat die Matrix folgende Gestalt:

$$A = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \tag{2.39}$$

Für einen kantengewichteten Graphen $G = (V, E, c)$ sind die Eintragungen der Adjazenzmatrix wie folgt:

$$a_{i,j} = \begin{cases} c_{i,j} & \text{, falls } (i,j) \in E \\ 0 & \text{, sonst} \end{cases} \tag{2.40}$$

2.3 Verwendete Datenbanken

2.3.1 MINT

In der literaturbasierten *MINT*-Datenbank [6], *Molecular INTeraction Database*, befindet sich eine Ansammlung von experimentell verifizierter Protein-Protein-Interaktionen, welche mit Hilfe dort angebotener Tools visualisiert werden können. Abbildung 2.4 zeigt eine solche Darstellung am Beispielprotein *SPOP*.

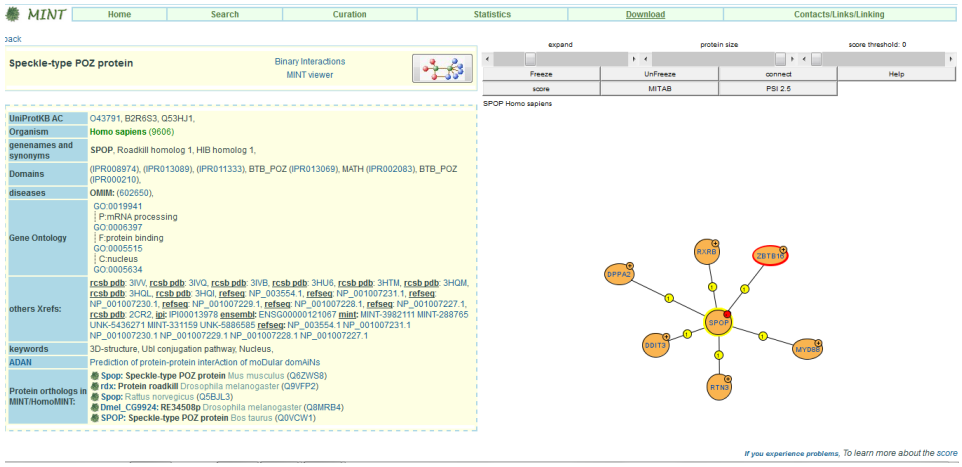


Abbildung 2.4: Beipielabfrage für das Protein *SPOP* an die *MINT*T-Datenbank

Diese Datenbank war eine der ersten Datenbanken, welche jeder Interaktion zwischen je zwei Proteinen einen Konfidenzwert zugewiesen hat. Zur Berechnung dieses Scores wurde im Jahre 2012 das „concept of integrated supporting evidence“eingeführt. Der Score $M \in [0,1]$ wird demnach wie folgt berechnet [6]:

$$M = 1 - c^{-y} \tag{2.41}$$

mit:

$$y = \sum_j S_j R_j \tag{2.42}$$

wobei $j \in \mathbb{N}$ die Anzahl der Eintragungen über die jeweilige Interaktion, $S \in [0,1]$ ein Maß für die Aussagekraft des zugrundeliegenden Experimentes

und $R \in [0,1]$ ein Maß für die Glaubwürdigkeit, welches von wissenschaftlichen Experten vergeben wird, ist. Einfluss auf den Parameter S haben hierbei die Art des Experimentes ($e \in [0,1]$), welches für die Nachweise der Interaktionen eingesetzt wurde. Wird beispielsweise ein Experiment verwendet, welches keine eindeutige Interaktion nachweist, so wird dem Wert e = 0.5 zugewiesen. Solch ein Experiment ist beispielsweise ein Pull-down-Experiment. Erfolgt durch ein Experiment ein eindeutiger Nachweis, so wird e der Wert 1 zugeordnet. Für S ergibt sich letztlich:

$$S = 1 - a^{-\sum_i e_i}, \tag{2.43}$$

wobei i die Anzahl der Experimente der jeweiligen Veröffentlichung ist und für a empirisch der Wert 1.2 gewählt wurde.
Bezüglich des Maßes für die Glaubwürdigkeit, so ergibt sich dieser aus folgender Berechnungsvorschrift:

$$R = 1 - b^{-r} \tag{2.44}$$

Dabei ist r die Anzahl der Zitierungen der Publikation und b wird wie a empirisch auf 1.2 gesetzt.

2.3.2 IntAct

Die *IntAct* [3] ist eine Datenbank, welche Informationen über molekulare Interaktionen enthält. Diese Daten stammen entweder aus Publikationen oder sind direkt in die Datenbank eingetragen worden. Wie bei der *MINT* besteht auch hier die Möglichkeit der grafischen Darstellung, wie Abbildung 2.5 zeigt.

Die *IntAct* bewertet ebenfalls Proteininteraktionen [3]. Dieser Wert wird *MI-Score* (S_{MI}) genannt. Berücksichtigt werden bei dieser Bewertung die Anzahl an Publikationen ($S_P \in [0,1]$), welche Informationen über die Interaktion liefert, die experimentelle Methode zum Interaktionsnachweis ($S_M \in [0,1]$),

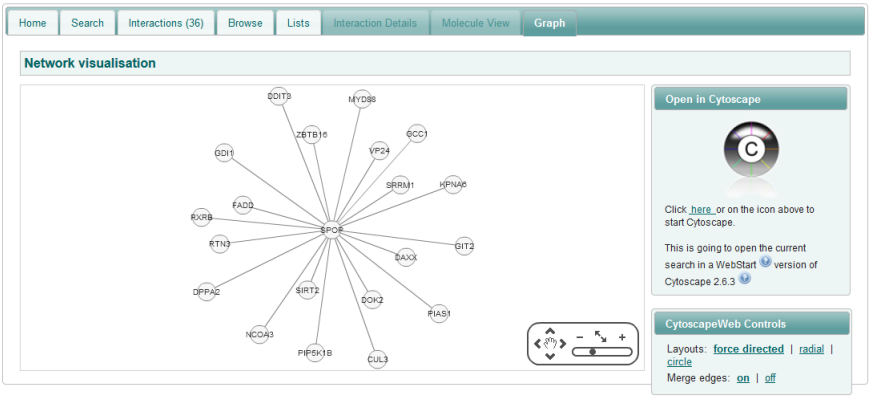


Abbildung 2.5: Beipielabfrage für das Protein *SPOP* an die *IntAct*-Datenbank

sowie der Interaktionstyp ($S_T \in [0,1]$). Der Publikationswert wird dabei wie folgt berechnet [3]:

$$S_P = \log_{b+1}(n+1) \tag{2.45}$$

mit $n \in \mathbb{N}$ die Anzahl der Publikationen, welche über die Interaktion berichten, $b \in \mathbb{N}$ die Anzahl der Publikationen, welche einen maximalen Score aufweisen. Ist kein solches b bekannt, so wird ihm 7 zugeordnet. Die Werte für die Experimente und der Interaktionsart ergeben sich dagegen aus folgender Berechnungsvorschrift:

$$S_{M,T} = \log_{b+1}(a+1) \quad \text{mit } a = \sum_i scv_i \cdot n_i,\ b = a + \sum_j max\{scv_j\}, \tag{2.46}$$

wobei scv_i definierte Werte sind, welche dem zugrundeliegendem Experiment bzw. dem nachgewiesenen Interaktionstyp zugeordnet werden.

$$S_{MI} = \frac{K_P \cdot S_P + K_M \cdot S_M + K_T \cdot S_T}{K_P + K_M + K_T} \tag{2.47}$$

mit $K_P,\ K_M,\ K_T,\ S_{MI} \in [0,1]$. Die Faktoren $K_{P,M,T}$ sind hierbei Wichtungsfaktoren, welche vom Anwender selber gewählt werden können. Erfolgt keine explizite Angabe dieser Wichtungsfaktoren, so wird ihnen automatisch der Wert 1 zugeordnet.

In beiden Datenbanken besteht die Möglichkeit, Ergebnisse einer Abfrage im *PSI-MI-TAB 2.6* Format zu speichern. Bei diesem Format handelt es sich um eine standartisiertes Format für die tabellarische Speicherung von Interaktionspartnern. Hierbei ist in jeder Zeile ein Interaktionspaar mit seinen zugehörigen Informationen gemäß der Spaltendefinition hinterlegt. Solche Spalten sind beispielsweise: eine eindeutige Identifikationsnummer für Interaktionsprotein A, eine eindeutige Identifikationsnummer für Interaktionsprotein B, Art des Experimentes, welches zum Nachweis verwendet wurde, Interaktionstyp und der zugehörige Konfidenzwert. Eine vollständige Beschreibung des Formates ist beschrieben in [4]

3 Analysemethoden im Protein-Protein Interaktions Netzwerk

3.1 Verknüpfung der Konfidenzwerte unterschiedlicher Datenbanken

Im vorherigen Abschnitt wurde bereits erwähnt, dass jede Datenbank ihre eigenen Berechnungsvorschriften verwendet, um eingetragene Interaktionen zwischen je zwei Proteinen zu bewerten. Ziel dieser Wertangabe ist es, gewissermaßen ein Maß für die „Vertrauenswürdigkeit“der Interaktion anzugeben. Möchte man jedoch aus den in der jeweiligen Datenbank befindlichen Proteinen ein gemeinsames Netzwerk unter Zuhilfenahme der angegeben Bewertungen erstellen, so ergeben sich folgende Probleme. Zum einen sind die verwendeten Bewertungsmethoden heuristische Verfahren mit unterschiedlichen Schwerpunkten und zum anderen unterscheidet sich die Anzahl an eingetragenen Interaktionspartnern in den Datenbanken sehr stark voneinander. Ein Beispiel, welche diese Problematik widerspiegelt, ist in Tabelle 3.1 gegeben. In der ersten Spalte befinden sich die Interaktionspartner zum Protein *SPOP*, welche in der *MINT*- und *IntAct*-Datenbank gefunden wurden. Die zweite und dritte Spalte der Tabelle enthalten die angegeben Scorewerte, wenn denn der Interaktionspartner in der jeweiligen Datenbank gelistet ist. Rot markierte Werte heben die Diskrepanz der unterschiedlichen Werte hervor.

Interaktionspartner	MI-Score	MINT-Score
MYD88	0.72	0.28
DAXX	0.69	-
NCOA3	0.63	-
CUL3	0.59	-
RXRB	0.55	0.28
SIRT2	0.40	-
GDI1	0.40	-
KPNA6	0.40	-
DOK2	0.40	-
PIP5K1B	0.40	-
SRRM1	0.37	-
DPPA2	0.37	0.28
GIT2	0.37	-
FADD	0.37	-
ZBTB16	0.37	0.28
DDIT3	0.37	0.28
RTN3	0.37	0.28
PIAS1	0.37	-

Tabelle 3.1: Zugehörigkeitswerte laut *MINT*- und *IntAct*-Datenbank für Beispiel *SPOP*

Ziel dieser Arbeit ist es nun, mit Hilfe der in Abschnitt 2.3.1-2.3.2 angegeben Datenbanken ein gemeinsames Interaktionsnetzwerk zu erstellen, wobei der Schwerpunkt hierbei darauf liegt, die verschiedenen Scorewerte miteinander zu verknüpfen.
Zunächst werden folgende Annahmen getroffen:

1. Die Scores der Interaktionen werden als „Zugehörigkeitswerte zum Interaktionsnetzwerk“ interpretiert.
2. Die Scores sind Elemente aus dem Einheitsintervall.
3. Der Wert „1 “ entspricht einer direkt nachgewiesenen Interaktion.
4. Der Wert „0 “ kann zwei verschiedene Bedeutungen haben:
   (a) Entweder enthält die jeweilige Datenbank keine Informationen über diese Interaktion oder

(b) es existiert keine Interaktion.

5. Ein höherer Score spiegelt entweder eine größere Anzahl an Informationen innerhalb der Datenbank wider oder die zugrundeliegenden Experimente, welche zum Nachweis verwendet wurden, sind hochwertiger.

Interpretiert man die Scores als Zugehörigkeitswerte, dann lassen sich die Annahmen 2-5 mathematisch wie folgt formulieren:

zu 2: $x_{i,j}^{DB_k}$ sei der Score laut Datenbank k, welcher der Interaktion zwischen Protein i und Protein j zugewiesen wird, mit $x_{i,j}^{DB_k} \in [0,1]$

zu 3: $1 \otimes x^{DB_k} = 1 \quad \forall x^{DB_k} \in [0,1]$ und $i,j,k \in \mathbb{N}$

zu 4: $0 \otimes x_{i,j}^{DB_k} = x_{i,j}^{DB_k} \quad \forall x_{i,j}^{DB_k} \in [0,1]$ und $i,j,k \in \mathbb{N}$

zu 5: $x_{i,j}^{DB_k} \otimes x_{i,j}^{DB_l} = x_{i,j}^{DB_k}$, falls $x_{i,j}^{DB_k} > x_{i,j}^{DB_l} \quad \forall x_{i,j}^{DB_k}, x_{i,j}^{DB_l} \in [0,1]$

Des Weiteren wird die Annahme getroffen, dass die Reihenfolge der Verknüpfung irrelevant ist:

$$\begin{aligned} x_{i,j}^{DB_k} \otimes x_{i,j}^{DB_l} &= x_{i,j}^{DB_l} \otimes x_{i,j}^{DB_k} \\ x_{i,j}^{DB_k} \otimes (x_{i,j}^{DB_l} \otimes x_{i,j}^{DB_m}) &= (x_{i,j}^{DB_k} \otimes x_{i,j}^{DB_l}) \otimes x_{i,j}^{DB_m} \end{aligned}$$

und

$$x_{i,j}^{DB_k} < x_{i,j}^{DB_l} \Rightarrow x_{i,j}^{DB_k} \otimes x_{i,j}^{DB_m} < x_{i,j}^{DB_l} \otimes x_{i,j}^{DB_m},$$

d.h. die gesuchte Verknüpfung soll kommutativ, assoziativ und monoton sein. Durch diese Annahmen kann man die Scores der verschiedenen Datenbanken mit Hilfe einer t-Conorm aus Abschnitt 2.2.2 verknüpfen. Durch Annahme 5 wird die Wahl der t-Conorm auf die Maximumnorm S_M beschränkt, wobei die Grundmenge X die Menge aller Proteine x darstellt. Weiterhin sei $\tilde{A}$ die Menge aller Interaktionspartner, die Ergebnis einer Abfrage einer Datenbank „A “sind, und $\mu_{\tilde{A}}(x)$ der Score, welcher als Zugehörigkeitswert des Proteins x zum Interaktionsnetzwerk interpretiert wird.

Führt man das Beispiel aus Tabelle 3.1 für das Protein *SPOP* fort, so ergeben sich neue Werte für die Zugehörigkeit, die in Tabelle 3.2 dargestellt sind.

Interaktionspartner	MI-Score	MINT-Score	S_M
MYD88	0.72	0.28	0.72
DAXX	0.69	0	0.69
NCOA3	0.63	0	0.63
CUL3	0.59	0	0.59
RXRB	0.55	0.28	0.55
SIRT2	0.40	0	0.40
GDI1	0.40	0	0.40
KPNA6	0.40	0	0.40
DOK2	0.40	0	0.40
PIP5K1B	0.40	0	0.40
SRRM1	0.37	0	0.37
DPPA2	0.37	0.28	0.37
GIT2	0.37	0	0.37
FADD	0.37	0	0.37
ZBTB16	0.37	0.28	0.37
DDIT3	0.37	0.28	0.37
RTN3	0.37	0.28	0.37
PIAS1	0.37	0	0.37

Tabelle 3.2: Verknüpfung der Zugehörigkeitswerte aus *MINT*- und *IntAct*-Datenbank für Beispiel *SPOP*

3.2 Wegealgorithmen für PPI-Netzwerke

Ein weiterer Schwerpunkt der Arbeit ist das Auffinden von Wegen in dem entstandenen PPI-Netzwerk, um perspektivisch mögliche Interaktionspartner herausfinden zu können. Unter einem Weg versteht man allgemein eine Folge von Knoten und Kanten, wobei jeder Knoten höchstens einmal im Weg enthalten sein darf. Die Proteine werden hierfür durch Knoten und Interaktionen zwischen je zwei Proteinen durch Kanten mit Kantengewicht repräsentiert. Die Kanten weisen hierbei keine Orientierung auf, d.h. es handelt sich um einen ungerichteten Graphen. Die gefundenen Wege können beispielsweise zum Visualisieren von Proteinnteraktionspartnern innerhalb des Netzwerkes dienen und können somit Auskunft über Besonderheiten zwischen Interaktionspartnern liefern. Um ein möglichst breites Anwendungsfeld zu ermöglichen, wurden zwei verschiedene Algorithmen bereitgestellt und implementiert. Möchte man beispielsweise einen kürzesten Weg zwischen je zwei Proteinen angeben, so ist hierfür der Dijkstra-Algorithmus zu verwenden. Da kürzeste Wege aber im Allgemeinen nur wenige Konten (Proteine) enthalten, können nicht alle Interaktionspartner dargestellt werden. Ergeben sich jedoch aus Anwendersicht Zielstellung, die nicht durch einen kürzesten Weg gelöst werden können, kann man den Algorithmus *BFS PPI-Network* verwenden, da dieser Algorithmus einen oder mehrere Wege zwischen je zwei Proteinen findet, deren Weglänge eine maximale Knotenanzahl nicht überschreitet. Nachfolgend werden beide Algorithmen erläutert.

3.2.1 Dijkstra-Algorithmus

Einer der wohl bekanntesten Algorithmen zum Finden von kürzesten Wegen ist der Edsger Dijkstra 1959 publizierte Dijkstra-Algorithmus [5]. Es handelt sich hierbei um ein Suchalgorithmus für Graphen, welcher das kürzeste-Wege-Problem für einen gegebenen Startknoten in einem Graphen löst. Die Kanten des Graphen können hierbei gewichtet sein. Relevant ist dabei nur, dass alle Kantengewichte positiv sind. Berechnet werden die kürzesten Wege

von einem Startknoten zu allen anderen Knoten des Graphen, wenn ein solcher existiert. Algorithmus 1 zeigt den Dijkstra-Algorithmus für einen Graphen mit Gewichtsfunktion und einem Startknoten s.

Algorithm 1 Dijkstra Algorithmus

Input: Graph $G = (V, E)$ **mit Gewichtsfunktion** $c : E \to \mathbb{R}^+$**, Startknoten** s

Output: kürzeste Wege $Paths$ **von** s **zu allen Knoten** $V \backslash \{s\}$

```
for all v ∈ V do
    distance(v) ← ∞
    visited(v) ← 0
    prev(v) ← 0
    Paths(v) ← 0
end for
distance(s) ← 0
Q ← s
while Q ≠ ∅ do
    find Node with Node ∈ Q, Node = min(distance), visited(Node) = 0
    delete Node in Q;
    visited(Node) ← 1
    for all v ∈ N[Node] do
        altpath ← distance(Node) + c(Node, v)
        if altpath < distance(v) and visited(v) = 0 then
            distance(v) ← altpath
            prev(v) ← Node
            Q ← v
        end if
    end for
end while
while prev(v) ≠ 0 do
    Paths ← Paths ∪ v
    v ← prev(v)
end while
```

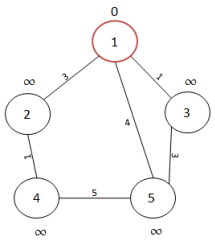
Zunächst werden für alle Knoten des Graphen die Distanzen mit ∞, eine Markierung $visited$ für bereits besuchte Knoten und die Liste $prev$, welche die Vorgängerknoten enthält mit 0 initialisiert. Die Distanz des Startknoten zu sich selber ist 0. Q ist eine Knotenmenge, in welcher alle noch nicht besuchten

Knoten enthalten sind. Aus diesem Grunde nennt man Q auch Warteschlange. Der erste Knoten dieser Menge ist der Startknoten. Nun wird der Knoten $Node \in Q$ bestimmt, welcher als noch unbesucht markiert ist und die geringste Distanz in $distance$ aufweist. Der Knoten wird aus der Warteschlange entfernt und als besucht gekennzeichnet. Anschließend werden für alle Nachbarknoten v von $Node$ die Distanzen berechnet. Ist diese berechnete Distanz kleiner als schon Vorhandene und ist der Nachbarknoten als noch nicht besucht markiert, so wird die Distanz aktualisiert und v der Warteschlange zugefügt. Der Algorithmus beginnt erneut mit dem ersten Knoten in der Warteschlange und terminiert, sobald Q keine Knoten mehr enthält. Anschließend können mittels $prev$ alle kürzesten Wege $Paths$ ermittelt werden. Ausgabe des Algorithmus sind alle kürzesten Wege ausgehend von einem Startknoten s

Die Laufzeit des Algorithmus beträgt dabei $\mathcal{O}(|V|^2)$, wobei $|V|$ die Knotenanzahl des Graphen ist.

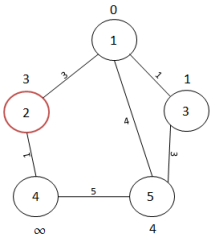
Beispiel für den Dijkstra-Algorithmus mit $s = 1$

Initialisierung:
$distance = (0, \infty, \infty, \infty, \infty)$,
$visited = (0, 0, 0, 0, 0)$,
$prev = (0, 0, 0, 0, 0)$,
$Q \leftarrow 1$



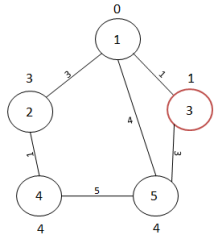
$Node \leftarrow 1,\ visited = (1, 0, 0, 0, 0),\ v = \{2, 3, 5\},$
$Q = \varnothing$

$v = 2 : altpath = 3,\ altpath < \infty,$
$distance(2) = 3, prev(2) = 1, Q = \{2\}$
$v = 3 : altpath = 1,\ altpath < \infty,$
$distance(3) = 1, prev(3) = 1, Q = \{2, 3\}$
$v = 5 : altpath = 4,\ altpath < \infty,$
$distance(5) = 4,\ prev(5) = 1,\ Q = \{2, 3, 5\},$
$distance = (0, 3, 1, \infty, 4)$



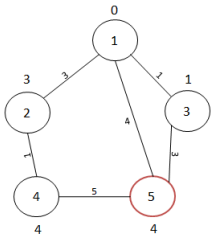
$Node \leftarrow 2,\ visited = (1, 1, 0, 0, 0),\ v = \{1, 4\},$
$Q = \{3, 5\}$

$v = 1,\ visited(1) = 1$
$v = 4 : altpath = 4,\ altpath < \infty,$
$distance(4) = 4,\ prev(4) = 2,\ Q = \{3, 5, 4\}$



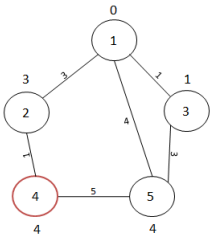
$Node \leftarrow 3,\ visited = (1, 1, 1, 0, 0),\ v = \{1, 5\},$
$Q = \{5, 4\}$

$v = 1 : visited(1) = 1$
$v = 5 : altpath = 4 = distance(5)$



$Node \leftarrow 5,\ visited = (1,1,1,0,1),\ v = \{1,3,4\},$
$Q = \{4\}$

$v = 1\ und\ v = 3\ ist\ visited = 1$
$v = 4,\ alt = 9 > distance(4)$



$Node \leftarrow 4,\ visited = (1,1,1,1,1),\ v = \{2,5\},$
$Q = \varnothing$

$v = 2\ und\ v = 5\ visited = 1$

Die Abbildungen 3.1 und 3.2 zeigen das Ergebnis des Dijkstra-Algorithmus zur Berechnung der kürzesten Wege zwischen allen Proteinen i und j. Als Grundlage dienten hierbei die Interaktionspartner aus den Datenbanken *MINT* und *IntAct*, welche zu dem Protein *STAT1* ermittelt werden konnten. Die Anzahl der Interaktionen laut *MINT* ist hierbei 13, bei der *IntAct* hingegen sind es 670. Diese Daten wurden zunächst mittels einer t-Conorm, der Maximumnorm, verknüpft. Dabei ergab sich ein Interaktionsnetzwerk mit 397 Proteinen. Diese Proteine wurden anschließend lexikografisch sortiert, so dass jeder Zahl auf der Ordinate und Abszisse im Bild eindeutig ein Protein zugeordnet werden kann. Für Abbildung 3.1 wurden die kürzesten Wege für das ungewichtete Netzwerk berechnet, wohingegen für Abbildung 3.2 das Netzwerk mit den Scores als Grundlage verwendet wurde.

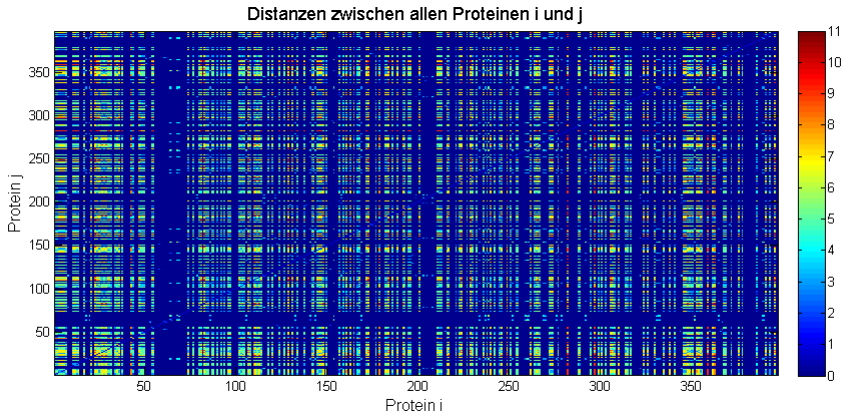


Abbildung 3.1: Distanzen zwischen allen Proteinen i, j im gewichteten Netzwerk

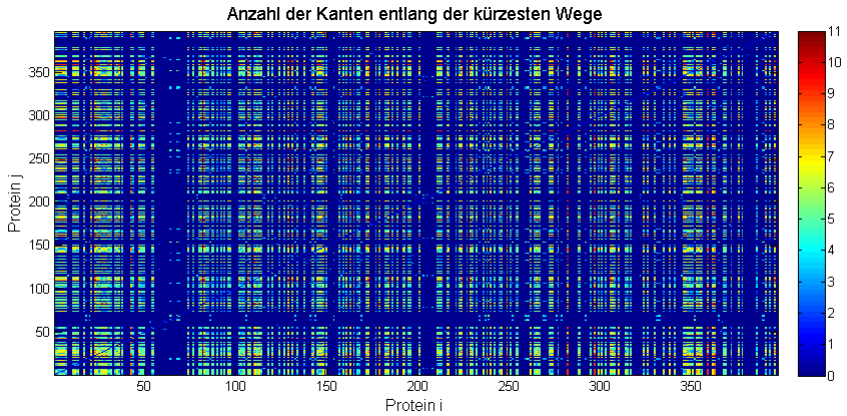


Abbildung 3.2: Anzahl der Kanten entlang der kürzesten Wege im ungewichteten Netzwerk

Ersichtlich ist, dass zwischen den beiden obigen Abbildungen kein signifikanter Unterschied besteht. So ergibt sich beispielsweise für den kürzesten Weg zwischen den Proteinen mit den Identifikationsnummern O00463 und P11207 eine Distanz von 8 im ungewichteten Netzwerk. Selbiges ergibt sich bezüglich der Kantenanzahl im gewichteten Graphen.

Die Darstellung 3.3 zeigt die Anzahl der kürzesten Wege in Abhängigkeit von der Anzahl der Kanten entlang des kürzesten Weges. Ersichtlich ist zunächst, dass trotz der großen Knotenanzahl die Anzahl der Kanten in den berechneten

kürzesten Wegen gering ist. Weiterhin ist erkennbar, dass eine Vielzahl der gefundenen kürzesten Wege eine Kantenanzahl von 5 aufweist. Einige wenige haben eine Kantenanzahl von 1, 2, 10 oder 11.

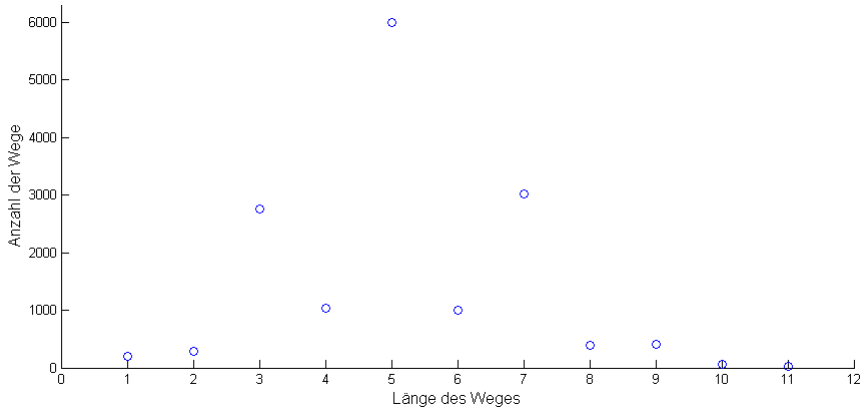


Abbildung 3.3: Anzahl der gefundenen Wege in Abhängigkeit der Weglänge

Ersichtlich ist zunächst, dass trotz der großen Knotenanzahl die Anzahl der Kanten in den berechneten kürzesten Wegen gering ist.

3.2.2 Breadth-First-Search

Ein Verfahren zum systematischen Durchsuchen von Graphen ist die Breitensuche (BFS) [5]. Hierbei werden zunächst vom Startknoten ausgehend alle seine Nachbarn betrachtet. Diese Nachbarn werden in eine Warteschlange eingefügt, so dass in jedem Iterationsschritt einer der Nachbarknoten vom Startknoten untersucht wird. Sind alle Nachbarn des Startknotens bearbeitet, so werden als Nächstes die Nachbarn der Nachbarn untersucht usw.. Das Verfahren terminiert, sobald die Warteschlange leer ist.

Im Rahmen dieser Arbeit wird ebenfalls ein Breitensuchverfahren verwendet, mit der Zielsetzung, alle Wege zwischen zwei gegeben Knoten zu finden. Hierbei werden nur Wege gesucht, deren Länge (Kantenanzahl) höchstens $n + \epsilon, n, \epsilon \in \mathbb{N}$ beträgt. Dabei entspricht n der Länge des kürzesten Weges zwischen je zwei Knoten, welcher der Algorithmus ermittelt und ϵ ist ein

vom Benutzer festgelegter Übergabeparameter. Der Algorithmus hierfür ist in Algorithmus 2 abgebildet und an einem Beispielgraphen, welcher aus vier Knoten und Kanten besteht, in Tabelle 3.3 gezeigt.

Algorithm 2 BFS PPI-Network

Input: Graph $G = (V, E)$**, Startknoten s, Terminalknoten t,** ϵ
Output: Wege $Paths$ **von s nach t deren Länge höchstens** $n + \epsilon$ **beträgt**

```
Qnode ← s
Qpaths ← s
Node ← s
Path ← s
visited ← 0
while Qnode, Qpaths ≠ ∅ and Length(Path) ≤ n + ϵ do

  for all v ∈ N[Node] do

    if v ∉ Path then

      if v == t then

        if visited == 0 then
          visited ← 1
          n ← lengthofPath
        end ifPaths ← Path ∪ v
      end if
      Qpaths ← Path ∪ v
      Qnode ← v
    end if
  end for
  delete Qpaths(1), Qnode(1)
  Qnode(1) ← Qnode(2)
  Qpaths(1) ← Qpaths(2)
  Node ← Qnode(1)
  Path ← Qpaths(1)
end while
```

Als Eingabe für den Algorithmus dient eine Adjazenzmatrix, welche den Graphen repräsentiert, ein Startknoten s, einen Zielknoten t und ein ϵ, welches vom Benutzer selbst gewählt werden kann. Gesucht sind alle Wege, deren

Länge höchstens $n + \epsilon$ beträgt. Zunächst werden die Warteschlangen $Qnode$ und $Qpaths$ sowie der aktuelle Knoten und Weg mit dem Startknoten initialisiert. Nun beginnt der eigentliche Algorithmus. Zunächst werden alle Nachbarn des Knoten $Node$ ermittelt und für jeden geprüft, ob er kein Element des aktuellen Weges $Path$ und ob er der Zielknoten ist. Sollte dem so sein, so werden die Knoten und der aktuelle Weg unter Hinzunahme der Knoten aus $Paths$, sowie aus der Warteschlange hinzugefügt. Ist ein Nachbarknoten kein Element des aktuell betrachteten Weges und nicht der Zielknoten, so wird v und $Path$ den Warteschlangen zugefügt. Sind alle Nachbarn des Knoten *Node* durchsucht, werden die ersten Listenelemente der Warteschlange entfernt und die Suche mit dem zweiten Element der Warteschlange beginnt von vorn. Der Algorithmus terminiert, wenn die Warteschlange leer und damit alle Knoten des Graphen untersucht sind.

Die Abbildungen 3.4 und 3.5 zeigen die Summe der Scores in Abhängigkeit von den Wegen, welcher der Algorithmus 2 ermittelt hat. Als Grundlage dienten auch hier, wie bereits für die Abbildunge 3.1 -3.3, die Interaktionspartner des Proteins *Stat1* aus den Datenbanken. Mit Hilfe des Algorithmus wurden alle Wege zwischen den Proteinen mit der Identifikationsnummer O00141 und Q05209, sowie O00401 und Q05209 bestimmt. Dabei ist $\epsilon = 5$ gewählt worden.

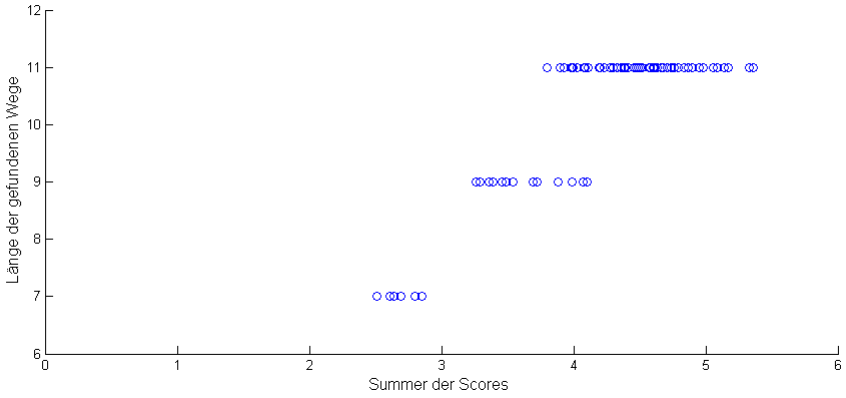


Abbildung 3.4: Wege zwischen Proteinen O00141 und Q05209

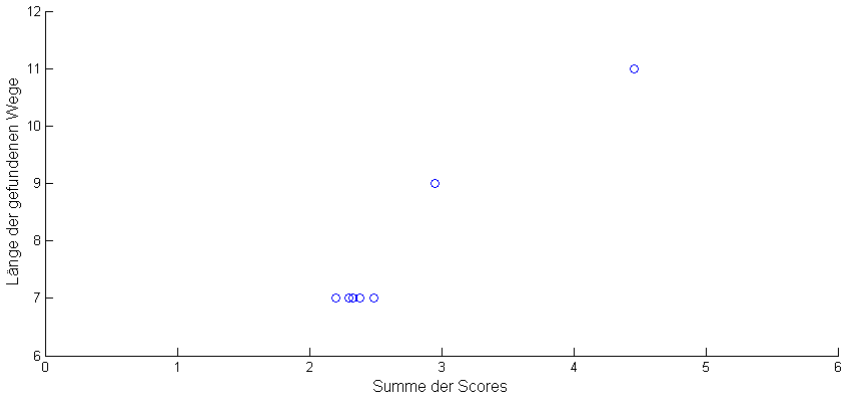
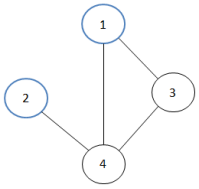


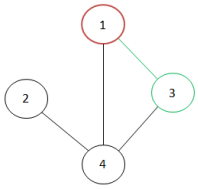
Abbildung 3.5: Wege zwischen Proteinen O00401 und Q05209

Für Abbildung 3.4 fand der Algorithmus 145 Wege, wobei der kürzeste Weg einen Summenscore von 2.51 aufwies. In Abbildung 3.5 hingegen fand der Algorithmus nur 7 Wege zwischen den beiden Proteinen mit einer minimalen Summe der Scores von 2.2 innerhalb des Netzwerkes.
Der BFS PPI-Network kann für viele verschiedene Zielstellungen eingesetzt werden. Grund hierfür ist, dass der Anwender selbst die Länge der Wege, in Abhängigkeit vom kürzesten Weg, bestimmen kann.

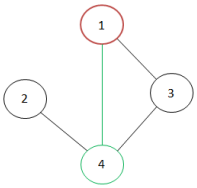
Beispiel für $|V| = 4$, $|E| = 4$, $s = 1$, $t = 2$ mit $\epsilon = 2$, $n = 4$



Initialisierung: $Qnode \leftarrow 1$, $Qpaths \leftarrow (1)$, $Node \leftarrow 1$, $Path \leftarrow (1)$, $N = \{3, 4\}$

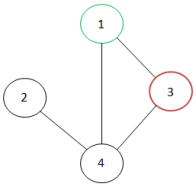


$v \leftarrow 3$, $v \notin Path$ und $3 \neq t$, $Qnode = \{1, 3\}$, $Qpaths = \{(1), (1, 3)\}$

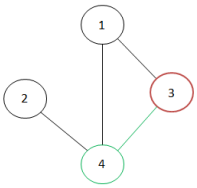


$v \leftarrow 4$, $v \notin Path$ und $4 \neq t$, $Qnode = \{1, 3, 4\}$, $Qpaths = \{(1), (1, 3), (1, 4)\}$

Da nun alle Nachbarn des Knotens 1 durchsucht wurden, wird anschließend der nachfolgende Knoten aus der Warteschlange $Qnode$ und der zugehörige Weg aus $Qpaths$ gewählt: $Node \leftarrow 3$, $Path \leftarrow (1, 3)$, $N = \{1, 4\}$

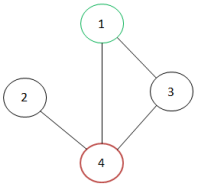


$v \leftarrow 1$, da jedoch $v \in Path$, wird mit dem nächsten Nachbarknoten fortgefahren



$v \leftarrow 4$, $v \notin Path$, $4 \neq t$, $Qnode = \{3, 4, 4\}$, $Qpaths = \{(1, 3), (1, 4), (1, 3, 4)\}$

$Node \leftarrow 4$, $Path \leftarrow (1, 4)$, $N = \{1, 2, 3\}$



$v \leftarrow 1$, $v \in Path$, Fortfahren mit nächsten Nachbarknoten

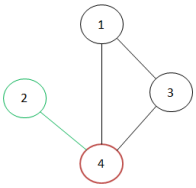
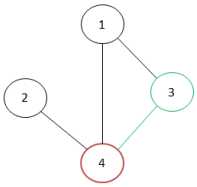
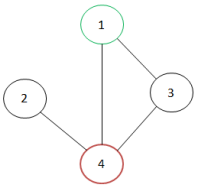
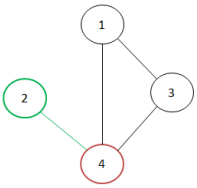
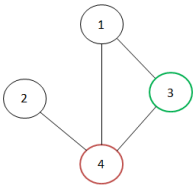
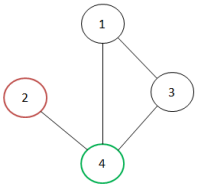
	$v \leftarrow 2$, $2 \notin Path$, $2 = t$, d.h. kürzeste Weg zwischen dem Startknoten 1 und dem Endknoten 2 ist gefunden, $Paths \leftarrow \{(1,4,2)\}$,$Qnode = \{4,4,2\}$, $Qpaths = \{(1,4),(1,3,4),(1,4,2)\}$
	$v \leftarrow 3$, $3 \notin Path$, $3 \neq t$, $Qnode = \{4,4,2,3\}$, $Qpaths = \{(1,4),(1,3,4),(1,4,2),(1,4,3)\}$ $Node \leftarrow 4$, $Path \leftarrow (1,3,4)$, $N = \{1,2,3\}$
	$v \leftarrow 1$, $v \in Path$
	$v \leftarrow 2$, $2 \notin Path$, $2 = t$, $Paths \leftarrow \{(1,3,4,2)\}$, $Qnode = \{4,2,3,2\}$, $Qpaths = \{(1,3,4),(1,4,2),(1,4,3),(1,3,4,2)\}$
	$v \leftarrow 3$, $v \in Path$ $Node \leftarrow 2$, $Path \leftarrow (1,4,2)$, $N = \{4\}$
	$v \leftarrow 4$, $v \in Path$ $Node \leftarrow 3$, $Path \leftarrow (1,4,3)$, $N = \{1,4\}$ $v \leftarrow 1$, $v \in Path$ $v \leftarrow 4$, $v \in Path$ $Node \leftarrow 2$, $Path \leftarrow (1,3,4,2)$, $N = \{4\}$ $v \leftarrow 4$, $v \in Path$ und damit ist sowohl $Qnode$ als auch $Qpaths$ leer und der Algorithmus terminiert.

Tabelle 3.3: Beispiel für den BFS-Algorithmus 2

4 Zusammenfassung und Diskussion

Das Erforschen von Protein-Protein-Interaktionen ist essentiell für das Verständnis von zellulären Vorgängen. Aus diesem Grunde existieren zahlreiche Datenbanken, die Informationen über PPI´s sammeln und speichern. Beispiele für solche Datenbanken sind die *MINT* und die *IntAct*. Diese Datenbaken enthalten zu den Interaktion nicht nur Information über verwendete Experimente, Publikation und Interaktionsart, sondern weisen diesen Interaktionen Konfidenzwerte zu, die jedoch aus unterschiedlichen Algorithmen hervorgehen. Ziel dieser Arbeit war es, diese verschiedenen Scores zu vereinigen. Es hat sich gezeigt, dass unter den Annahmen aus Abschnitt 3.1 eine t-Conorm angewendet werden kann. Es wurde hierbei die Maximumnorm gewählt. Anschließend ist mit Hilfe der Datenbankinformationen ein Graph $G = (V, E)$ entstanden, wobei V die Menge aller Proteine ist und zwei Knoten adjazent sind, wenn eine Interaktion zwischen ihnen stattfindet. Die aus der t-Conorm entstandenen Werte dienten dabei als Kantengewichte. Für diese Graphen sind zwei Algorithmen angegeben worden, mit dessen Hilfe sich Wege innerhalb des Netzwerkes berechnen lassen. Algorithmus 1 dient hierbei dem Bestimmen kürzester Wege von einem gegeben Startknoten zu allen weiteren Knoten des Graphen. Der Algorithmus 2 verwendet ein Breitensuchverfahren, um im Graphen systematisch alle Wegen zwischen zwei Proteinen zu berechnen. Da es sich bei Proteininteraktionsnetzwerken i.A. um große Netzwerke handelt, wurde eine obere Schranke für die Länge der Wege angegeben mit $n + \epsilon$. Der Parameter ϵ kann dabei vom Anwender selbst festgelegt werden.
Die Aufgabe Konfidenzwerte aus Datenbanken für Proteininteraktionen sinnvoll zu verknüpfen, liefert ein breites Spektrum für neue Ansätze und Weiterentwicklungen. Zunächst einmal scheint es sinnvoll Informationen weitere Datenbanken einzubinden. Problematisch hierbei ist, dass nicht notwendig alle Scores Elemente aus dem Einheitsintervall sind. Eine Datenbank, deren Werte jedoch ebenfalls Elemente aus dem Einheitsintervall sind, ist die *STRING*-Datenbank [11].

Ein weiterer Ansatzpunkt für Weiterentwicklungen bezüglich der Verknüpfung von Scores wäre die Anwendung von parametrisierten t-Cornomen. Die Grundidee dieser Parametrisierung ist es, einen Operator zu konstruieren, der so viele t-Conormen wie nur möglich umfasst. Parametrisierte t-Conormen enthalten daher einen Parameter γ, welcher beliebig positive reellwertige Zahlen annehmen kann. Ein Beipiel für solch einen Operator ist der Hammacher-Operator [8] und ist wie folgt für t-Conormen definiert:

$$H_{\gamma}^{S} = \frac{\mu_{\tilde{A}}(x) + \mu_{\tilde{B}}(x) - (2-\gamma) \cdot \mu_{\tilde{A}}(x) \cdot \mu_{\tilde{B}}(x)}{1 - (1-\gamma) \cdot \mu_{\tilde{A}}(x) \cdot \mu_{\tilde{B}}(x)} \tag{4.1}$$

mit $\gamma \in [0, 1]$. Ist $\gamma = 1$ so ergibt sich hieraus das algebraische Produkt aus Abschnitt 2.2.2. Weitere solche Operatoren wären der *Frank*-Operator, der *Weber*-Operator und der *Yager*-Operator, um nur einige an dieser Stelle zu nennen. Anschließend könnte untersucht werden, inwiefern sich dabei die Wege im Graphen von den hier gefundenen Wegen ändern.
Ebenfalls einbezogen werden könnte die Theorie der Fuzzy Markov Ketten [1]. Diese unterscheiden sich von Markov Ketten insofern, dass die Einträge der Übergangsmatrix keine Wahrscheinlichkeiten sondern Möglichkeiten sind und bei der Übergangsmatrix T handelt es sich um keine stochastische Matrix. D.h. die Summer der Zeilenelemente müssen nicht „1“sein, sondern in jeder Zeile muss ein Element existieren mit $(t_{i,j}) = 1$. Wäre es also möglich eine geeignete Möglichkeitsverteilung für das hier beschriebene Problem zu finden, so könnte mit Hilfe der Übergangsmatrix Aussagen über mögliche „Wege“innerhalb des Netzwerkes gemacht werden.

Literaturverzeichnis

[1] Eslami Esfandiar Buckley, James J. An Introduction to Fuzzy Logic and Fuzzy Sets.

[2] Oliver Deiser. Einführung in die Mengenlehre Die Mengenlehre Georg Cantors und ihre Axiomatisierung durch Ernst Zermelo. Springer Berlin Heidelberg, 2010.

[3] Samuel Kerrien, Bruno Aranda, Lionel Breuza, Alan Bridge, Fiona Broackes-Carter, Carol Chen, Margaret Duesbury, Marine Dumousseau, Marc Feuermann, Ursula Hinz, Christine Jandrasits, Rafael C. Jimenez, Jyoti Khadake, Usha Mahadevan, Patrick Masson, Ivo Pedruzzi, Eric Pfeiffenberger, Pablo Porras, Arathi Raghunath, Bernd Roechert, Sandra E. Orchard, and Henning Hermjakob. The intact molecular interaction database in 2012. Nucleic Acids Research, 40(Database-Issue):841–846, 2012.

[4] Samuel Kerrien, Sandra Orchard, Luisa Montecchi-Palazzi, Bruno Aranda, Antony F. Quinn, Nisha Vinod, Gary D. Bader, Ioannis Xenarios, Jérôme Wojcik, David Sherman, Mike Tyers, John J. Salama, Susan Moore, Arnaud Ceol, Andrew Chatr-Aryamontri, Matthias Oesterheld, Volker Stümpflen, Lukasz Salwinski, Jason Nerothin, Ethan Cerami, Michael E. Cusick, Marc Vidal, Michael Gilson, John Armstrong, Peter Woollard, Christopher Hogue, David Eisenberg, Gianni Cesareni, Rolf Apweiler, and Henning Hermjakob. Broadening the horizon–level 2.5 of the hupo-psi format for molecular interactions. BMC biology, 5(1):44+, October 2007.

[5] SvenOliver Krumke and Hartmut Noltemeier. Graphentheoretische Konzepte und Algorithmen. Vieweg+Teubner, 2009.

[6] Luana Licata, Leonardo Briganti, Daniele Peluso, Livia Perfetto, Marta Iannuccelli, Eugenia Galeota, Francesca Sacco, Anita Palma, Aurelio Nardozza, Elena Santonico, Luisa Castagnoli, and Gianni Cesareni. Mint, the molecular interaction database: 2012 update. Nucleic Acids Research, 40(Database-Issue):857–861, 2012.

[7] W.M. Lippe. Soft-Computing: Mit Neuronalen Netzen, Fuzzy-Logic und Evolutionären Algorithmen. Springer London, Limited, 2007.

[8] Andrzej Piegat. Fuzzy Modeling and Control. Physica Verlag Heidelberg, 1 edition, 2001.

[9] Prof. Dr. med. Dr. sc. Dr. h. c. Jürgen Roth Prof. Dr. med. Margit Pavelka. Funktionelle Ultrastruktur Atlas der Biologie und Pathologie von Geweben. SpringerWienNewYork, 1 edition, 2005.

[10] A.P. Read, D. Donnai, O. Riess, J. Zschocke, U.A. Mau-Holzmann, and S. Kuhlmann-Krieg. Angewandte Humangenetik. Walter De Gruyter Incorporated, 2008.

[11] Christian von Mering, Lars Juhl Jensen, Berend Snel, Sean D. Hooper, Markus Krupp, Mathilde Foglierini, Nelly Jouffre, Martijn A. Huynen, and Peer Bork. String: known and predicted protein-protein associations, integrated and transferred across organisms. Nucleic Acids Research, 33(Database-Issue):433–437, 2005.

[12] Ioannis Xenarios and David Eisenberg. Protein interaction databases. Current Opinion in Biotechnology, 12(4):334 – 339, 2001.

[13] L.A. Zadeh. Fuzzy sets. Information and Control, 8(3):338 – 353, 1965.

[14] Aidong Zhang. Protein interaction networks : computational analysis. Cambridge University Press, 1 edition, April 2009.

Selbständigkeitserklärung

Hiermit erkläre ich, dass ich meine Arbeit selbstständig verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und die Arbeit noch nicht anderweitig für Prüfungszwecke vorgelegt habe. Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht

Mittweida, 01.12.2013